REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** 8/6/98	**3. REPORT TYPE AND DATES COVERED** Final 6/1/96 - 5/31/98

4. TITLE AND SUBTITLE
Algorithms and Tools for the Automatic Analysis of Embedded Systems

5. FUNDING NUMBERS
N00014-96-1-0883

6. AUTHOR(S)
T.A. Henzinger

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of California
Berkeley, CA 94720

8. PERFORMING ORGANIZATION REPORT NUMBER
442427-23113

9. SPNSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Ballston Centre Tower One
800 Norht Quincy Street
Arlington, VA 22217-5660

10. SPONSORING /MONITORING AGENCY
Office of Naval Research-Seattle Regional Office

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Over the duration of the award, we developed formal methods and tools for the design, verification, and control of embedded systems. This was accomplished by extending techniques, based on automata theory and temporal logic, from the analysis of time-independent reactive systems to real-time embedded systems. As system specification language for embedded systems, we introduced hybrid automata, which equip traditional discrete automata with real-numbered clock variables and continuous environment variables. As requirements specification, we introduced temporal logics with clock variables for expressing timing constraints.

14. SUBJECT TERMS
model checking of time and hybrid systems

15. NUMBER OF PAGES
8

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT**

ONR Grant N00014-96-1-0883

Algorithms and Tools for the Automatic Analysis of Embedded Systems

Thomas A. Henzinger
Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720-1770
Phone: (510) 643-2430
Fax: (510) 643-5052
Email: tah@eecs.berkeley.edu

1 Final Report: 1 June 1995 – 31 May 1998

Summary of Completed Project

We developed formal methods and tools for the design, verification, and control of embedded systems. This was accomplished by extending techniques, based on automata theory and temporal logic, from the analysis of time-independent reactive systems to real-time embedded systems. As system specification language for embedded systems, we introduced hybrid automata, which equip traditional discrete automata with real-numbered clock variables and continuous environment variables. As requirements specification languages, we introduced temporal logics with clock variables for expressing timing constraints.

Since the state spaces of systems with real-numbered clock variables are infinite, all verification must proceed symbolically. Symbolic verification methods are based either on deductive reasoning, using proof rules for symbolic logics, or on algorithmic analysis, using model checking procedures that operate on symbolic representations of state sets. We developed proof calculi for checking if a hybrid automaton satisfies linear-time clock properties, and we developed and implemented symbolic procedures for checking if a piecewise-linear hybrid automaton satisfies branching-time clock properties. The continuous variables of piecewise-linear hybrid automata follow trajectories within piecewise-linear envelopes, which can be used to approximate conservatively the behavior of more general, nonlinear systems.

Our algorithms have been implemented in the tool HyTech, the first model checker for hybrid (mixed discrete-continuous) systems. We applied HyTech to benchmark case studies in railway control, steam-boiler control, manufacturing-system and audio-system control. We also established the theoretical complexity of various formulations of the verification and control problems for hybrid systems, and we identified the exact boundary between decidability and undecidability of real-time reasoning.

Deliverables

PhD Dissertations

Pei-Hsin Ho (1995): *Automatic Analysis of Hybrid Systems*
Peter W. Kopke (1996): *The Theory of Rectangular Hybrid Automata*

Publications Detailed explanations of research accomplishments and publications up to 5/31/97 can be found in previous annual reports. Our research accomplishments and publications since 6/1/97 are listed below. All publications can be downloaded from

http://www.eecs.berkeley.edu/~tah/Publications

Software We developed and implemented HYTECH, a symbolic model checker for the automatic analysis of embedded real-time systems. HYTECH has around 100 registered users worldwide, and has been applied, by other researchers, in such diverse fields as the design and analysis of automotive systems (engine control [1], suspension control [2]), aeronautics systems (collision avoidance [3], landing gear control [4]), chemical engineering [5], and robotics [6]. HYTECH, together with usage information, can be downloaded from

http://www.eecs.berkeley.edu/~tah/HyTech

[1] T. Villa, H. Wong-Toi, A. Balluchi, J. Preußig, A. Sangiovanni Vincentelli, and Y. Watanabe, "Formal verification of an automotive engine controller in cutoff mode," 1998, submitted.

[2] T. Stauner, O. Müller, and M. Fuchs, "Using HYTECH to verify an automotive control system," *HART 97: Hybrid and Real-time Systems*, Lecture Notes in Computer Science 1201, Springer-Verlag, 1997, 139–153.

[3] J. Kosecka, C.J. Tomlin, G. Pappas, and S. Sastry, "Generation of conflict resolution manoeuvres for air traffic management," *IROS 97: IEEE/RSJ International Conference on Intelligent Robot and Systems*, Vol. 3, IEEE, 1997, 1598–1603.

[4] S. Nadjm-Tehrani and J.-E. Strömberg, "Proving dynamic properties in an aerospace application," *RTSS 95: Real-time Systems Symposium*, IEEE Computer Society Press, 1995, 2–10.

[5] O. Stursberg, S. Kowalewski, I. Hoffmann, and J. Preußig, "Comparing timed and hybrid automata as approximations of continuous systems," *Hybrid Systems IV*, Lecture Notes in Computer Science 1273, Springer-Verlag, 1997, 361–377.

[6] J.C. Corbett, "Timing analysis of Ada tasking programs," *IEEE Transactions on Software Engineering* 22(7):461–483, 1996.

2 Third Annual Progress Report: 1 June 1997 – 31 May 1998

Abstract

Over the past year, we made both theoretical and practical progress regarding the specification, verification, and control of embedded systems. The highlights we achieved on the theoretical side include the solution of a longstanding open question as to which real-time specification formalisms are *regular* (i.e., well-behaved: closed under boolean operations, decidable, and robust), and we invented the first automatic synthesis procedure for controllers of hybrid systems which are not strictly real-time systems (i.e., they include more general behavior than what can be described by clocks). The highlights we achieved on the practical side include an efficient algorithm for the *approximate* analysis of hybrid systems, and a successful combination of the model checker HYTECH with the theorem prover PVS in a case study. Much of the past year was also spent on summarizing and disseminating the ideas that were developed during the course of this project; for example, we wrote invited tutorial introductions on our approach to real-time systems [A1], on model checking timing constraints [A4], on model checking hybrid systems [B1], and on HYTECH [B2].

Keywords

Modeling, verification, and control of real-time and hybrid systems.

Report

We made progress in the following two directions.

Real-time systems In [A1], we wrote a tutorial introduction to the clock approach for designing real-time systems, which we pioneered. In [A2], we solved a longstanding open question by classifying the regular real-time formalisms. In [A3], we provided a complete axiomatization of the regular real-time specification logics. In [A4], we summarized the decision procedures for the regular real-time specification logics, which we developed earlier in the course of this project. In [A5], we wrote the complete journal version of an earlier paper on reasoning about accumulated delays in real-time systems.

Hybrid systems In [B1], we wrote a tutorial introduction to the symbolic approach for verifying and controlling hybrid systems, which we pioneered. In [B2], we summarized the tool HyTech and its applications, which we developed in the course of this project. In [B3], we presented two methods for approximating nonlinear hybrid systems using piecewise-linear and rectangular behavior. In [B4], we combined model-checking methods based on HyTech with theorem-proving methods based on Pvs for the verification of hybrid systems. In [B5], we introduced a new, more efficient algorithm for the analysis of the important class of rectangular hybrid systems. In [B6], we showed that sampling controllers for rectangular hybrid systems can be synthesized automatically. In [B7], we extended the model of hybrid automata to permit the compositional description and design of embedded systems.

Our specific accomplishments are described below.

A Models and Logics for Real-time Systems

[A1] Rajeev Alur, Thomas A. Henzinger, "Real-time system = discrete system + clock variables," *Software Tools for Technology Transfer* **1**, 1997, pp. 86–109. (Also appeared as Technical Report UCB/ERL-M97/78, University of California at Berkeley, October 1997.)

Programs such as device drivers and embedded controllers must explicitly refer and react to time. For this purpose, a variety of language constructs—including delays, timeouts, and watchdogs—have been put forward. We advocate an alternative proposal, namely, to designate certain program variables as clock variables. The value of a clock variable changes as time advances. Timing constraints can be expressed, then, by conditions on clock values. A single new language construct—the guarded wait statement—suffices to enforce the timely progress of a program. We illustrate the use of clock variables and guarded wait statements with real-time applications such as round-robin (timeout-driven) and priority (interrupt-driven) scheduling. Clock variables generalize naturally to variables that measure environment parameters other than time. This observation leads to a language for hybrid (mixed digital-analog) applications such as embedded process control.

This paper introduces, gently but rigorously, the clock approach to real-time programming. We present with mathematical precision, assuming no prerequisites other than familiarity with logical and programming notations, the concepts that are necessary for understanding, writing, and executing clock programs. In keeping with an expository style, all references are clustered in bibliographic remarks at the end of each section. The first appendix presents proof rules for verifying temporal properties of clock programs. The second appendix points to selected literature on formal methods and tools for programming with clocks. In particular, the timed automaton, which is a finite-state machine equipped with clocks, has become a standard paradigm for real-time model checking; it underlies the tools HyTech, Kronos, and Uppaal, discussed elsewhere in this volume.

[A2] Thomas A. Henzinger, J.-F. Raskin, P.-Y. Schobbens, "The regular real-time languages," *Proceedings of the 25th International Colloquium on Automata, Languages, and Programming* (ICALP 98), *Lecture Notes in Computer Science* **1443**, Springer-Verlag, 1998, pp. 580–591.

A specification formalism for reactive systems defines a class of ω-languages. We call a specification formalism *fully decidable* if it is constructively closed under boolean operations and has a decidable satisfiability (nonemptiness) problem. There are two important, robust classes of ω-languages that are definable by fully decidable formalisms. The *ω-regular languages* are definable by finite automata, or equivalently, by the Sequential Calculus. The *counter-free ω-regular languages* are definable by temporal logic, or equivalently, by the first-order fragment of the Sequential Calculus. The gap between both classes can be closed by finite counting (using automata connectives), or equivalently, by projection (existential second-order quantification over letters).

A specification formalism for real-time systems defines a class of *timed* ω-languages, whose letters have real-numbered time stamps. Two popular ways of specifying timing constraints rely on the use of clocks, and on the use of time bounds for temporal operators. However, temporal logics with clocks or time bounds have undecidable satisfiability problems, and finite automata with clocks (so-called *timed automata*) are not closed under complement. Therefore, two fully decidable restrictions of these formalisms have been proposed. In the first case, clocks are restricted to *event clocks*, which measure distances to immediately preceding or succeeding events only. In the second case, time bounds are restricted to *nonsingular intervals*, which cannot specify the exact punctuality of events. We show that the resulting classes of timed ω-languages are robust, and we explain their relationship.

First, we show that temporal logic with event clocks defines the same class of timed ω-languages as temporal logic with nonsingular time bounds, and we identify a first-order monadic theory that also defines this class. Second, we show that if the ability of finite counting is added to these formalisms, we obtain the class of timed ω-languages that are definable by finite automata with event clocks, or equivalently, by a restricted second-order extension of the monadic theory. Third, we show that if projection is added, we obtain the class of timed ω-languages that are definable by timed automata, or equivalently, by a richer second-order extension of the monadic theory. These results identify three robust classes of timed ω-languages, of which the third, while popular, is not definable by a fully decidable formalism. By contrast, the first two classes are definable by fully decidable formalisms from temporal logic, from automata theory, and from monadic logic. Since the gap between these two classes can be closed by finite counting, we dub them the *timed ω-regular languages* and the *timed counter-free ω-regular languages*, respectively.

[A3] J.-F. Raskin, P.-Y. Schobbens, Thomas A. Henzinger, "Axioms for real-time logics," *Proceedings of the Ninth International Conference on Concurrency Theory* (CONCUR 98), *Lecture Notes in Computer Science*, Springer-Verlag, 1998.

This paper presents a complete axiomatization of fully decidable propositional real-time linear temporal logics with past: the Event Clock Logic (ECL) and the Metric Interval Temporal Logic with past (MITL). The completeness proof consists of an effective proof building procedure for ECL. From this result we obtain a complete axiomatization of MITL by providing axioms translating MITL formulae into ECL formulae, the two logics being equally expressive. Our proof is structured to yield a similar axiomatization and procedure for interesting fragments of these logics, such as the linear temporal logic of the real numbers (LTR).

[A4] Thomas A. Henzinger, "It's about time: real-time logics reviewed," *Proceedings of the Ninth International Conference on Concurrency Theory* (CONCUR 98), *Lecture Notes in Computer Science*, Springer-Verlag, 1998. (Also available as Technical Report UCB/ERL-M98/40, University of California at Berkeley, June 1998.)

We summarize and reorganize some of the last decade's research on real-time extensions of temporal logic. Our main focus is on tableau constructions for model checking linear temporal formulas with timing constraints. In particular, we find that a great deal of real-time verification can be performed in polynomial space, but also that considerable care must be exercised in order to keep the real-time verification problem in polynomial space, or even decidable.

[A5] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, "Computing accumulated delays in real-time systems," *Formal Methods in System Design* **11**, 1997, pp. 137–156. (Also appeared as Technical Report UCB/ERL-M97/19, University of California at Berkeley, March 1997.)

We present a verification algorithm for duration properties of real-time systems. While simple real-time properties constrain the total elapsed time between events, duration properties constrain the accumulated satisfaction time of state predicates. We formalize the concept of durations by introducing duration measures for timed automata. A duration measure assigns to each finite run of a timed automaton a real number —the *duration* of the run— which may be the accumulated satisfaction time of a state predicate along the run. Given a timed automaton with a duration measure, an initial and a final state, and an arithmetic constraint, the *duration-bounded reachability problem* asks if there is a run of the automaton from the initial state to the final state such that the duration of the run satisfies the constraint. Our main result is an (optimal) PSPACE decision procedure for the duration-bounded reachability problem.

B Verification and Control of Hybrid Systems

[B1] Rajeev Alur, Thomas A. Henzinger, Howard Wong-Toi, "Symbolic analysis of hybrid systems," *Proceedings of the 36th Annual IEEE Conference on Decision and Control* (CDC 97), December 1997, pp. 702–707. (Also available as Technical Report UCB/ERL-M98/23, University of California at Berkeley, April 1998.)

A *hybrid system* is a dynamical system whose behavior exhibits both discrete and continuous change. A *hybrid automaton* is a mathematical model for hybrid systems, which combines, in a single formalism, automaton transitions for capturing discrete change with differential equations for capturing continuous change. In this survey, we demonstrate symbolic algorithms for the verification and controller synthesis of *linear hybrid automata*, a subclass of hybrid automata that can be analyzed automatically.

[B2] Thomas A. Henzinger, Pei-Hsin Ho, Howard Wong-Toi, "HyTech: a model checker for hybrid systems," *Software Tools for Technology Transfer* **1**, 1997, pp. 110–122. (Also appeared as Technical Report UCB/ERL-M97/79, University of California at Berkeley, October 1997.) Preliminary version appeared in *Proceedings of the Ninth International Conference on Computer-aided Verification* (CAV 97), *Lecture Notes in Computer Science* **1254**, Springer-Verlag, 1997, pp. 460–463.

A hybrid system consists of a collection of digital programs that interact with each other and with an analog environment. Examples of hybrid systems include medical equipment, manufacturing controllers, automotive controllers, and robots. The formal analysis of the mixed digital-analog nature of these systems requires a model that incorporates the discrete behavior of computer programs with the continuous behavior of environment variables, such as temperature and pressure. Hybrid Automata capture both types of behavior by combining finite automata with differential inclusions (i.e. differential inequalities). HyTech is a symbolic model checker for linear hybrid automata, an expressive, yet automatically analyzable, subclass of hybrid automata. A key feature of HyTech is its ability to perform parametric analysis, i.e. to determine the values of design parameters for which a linear hybrid automaton satisfies a temporal requirement.

[B3] Thomas A. Henzinger, Pei-Hsin Ho, Howard Wong-Toi, "Algorithmic analysis of nonlinear hybrid systems," *IEEE Transactions on Automatic Control* **43**, 1998, pp. 540–554.

Hybrid systems are digital real-time systems that are embedded in analog environments. Model-checking tools are available for the automatic analysis of *linear hybrid automata*, whose environment variables are subject to piecewise-constant polyhedral differential inclusions. In most embedded systems, however, the environment variables have differential inclusions that vary with the values of the variables, e.g., $\dot{x} = x$. Such inclusions are prohibited in the linear hybrid automaton model. We present two methods for translating nonlinear hybrid systems into linear hybrid automata. Properties of the nonlinear systems can then be inferred from the automatic analysis of the translated linear hybrid automata.

The first method, called *clock translation*, replaces constraints on nonlinear variables by constraints on clock variables. The clock translation is efficient but has limited applicability. The second method, called *linear phase-portrait approximation*, conservatively overapproximates the phase portrait of a hybrid automaton using piecewise-constant polyhedral differential inclusions. Both methods are sound for safety properties; that is, if we establish a safety property of the translated linear system, we may conclude that the original nonlinear system satisfies the property. When applicable, the clock translation is also complete for safety properties; that is, the original system and the translated system satisfy the same safety properties. The phase-portrait approximation method is not complete for safety properties, but it is asymptotically complete;

intuitively, for every safety property, and for every relaxed nonlinear system arbitrarily close to the original, if the relaxed system satisfies the safety property, then there is a linear phase-portrait approximation that also satisfies the property.

We illustrate both methods by using HyTech—a symbolic model checker for linear hybrid automata—to automatically check properties of a nonlinear temperature controller and of a predator-prey ecology.

[B4] Thomas A. Henzinger, Vlad Rusu, "Reachability verification for hybrid automata," *Proceedings of the First Workshop on Hybrid Systems: Computation and Control* (HSCC 98), *Lecture Notes in Computer Science* **1386**, Springer-Verlag, 1998, pp. 190–204. (Also available as Technical Report UCB/ERL-M98/19, University of California at Berkeley, April 1998.)

We study the reachability problem for hybrid automata. Automatic approaches, which attempt to construct the reachable region by symbolic execution, often do not terminate. In these cases, we require the user to guess the reachable region, and we use a theorem prover (Pvs) to verify the guess. We classify hybrid automata according to the theory in which their reachable region can be defined finitely. This is the theory in which the prover needs to operate in order to verify the guess. The approach is interesting, because an appropriate guess can often be deduced by extrapolating from the first few steps of symbolic execution.

[B5] Jörg Preußig, Stephan Kowalewski, Howard Wong-Toi, Thomas A. Henzinger, "An algorithm for the approximative analysis of rectangular automata," to appear in the *Proceedings of the Fifth International Symposium on Formal Techniques in Real-time and Fault-tolerant Systems* (FTRTFT 98), *Lecture Notes in Computer Science*, Springer-Verlag, 1998.

Rectangular automata are well suited for approximate modeling of continuous-discrete systems. The exact analysis of these automata is feasible for small examples but can encounter severe numerical problems for even medium-sized systems. This paper presents an analysis algorithm that uses conservative overapproximation to avoid these numerical problems. The algorithm is demonstrated on a simple benchmark system consisting of two connected tanks.

[B6] Thomas A. Henzinger, Peter W. Kopke, "Discrete-time control for rectangular hybrid automata," *Proceedings of the 24th International Colloquium on Automata, Languages, and Programming* (ICALP 97), *Lecture Notes in Computer Science* **1256**, Springer-Verlag, 1997, pp. 582–593. (Also available as Technical Report UCB/ERL-M97/29, University of California at Berkeley, April 1997.)

Rectangular hybrid automata model digital control programs of analog plant environments. We study rectangular hybrid automata where the plant state evolves continuously in real-numbered time, and the controller samples the plant state and changes the control state discretely, only at the integer points in time. We prove that rectangular hybrid automata have finite bisimilarity quotients when all control transitions happen at integer times, even if the constraints on the derivatives of the variables vary between control states. This is sharply in contrast with the conventional model where control transitions may happen at any real time, and already the reachability problem is undecidable. Based on the finite bisimilarity quotients, we give an exponential algorithm for the symbolic sampling- controller synthesis of rectangular automata. We show our algorithm to be optimal by proving the problem to be EXPTIME-hard. We also show that rectangular automata form a maximal class of systems for which the sampling-controller synthesis problem can be solved algorithmically.

[B7] Rajeev Alur, Thomas A. Henzinger, "Modularity for timed and hybrid systems," *Proceedings of the Eighth International Conference on Concurrency Theory* (CONCUR 97), *Lecture Notes in Computer Science* **1243**, Springer-Verlag, 1997, pp. 74–88.

In a trace-based world, the modular specification, verification, and control of live systems require each module to be receptive; that is, each module must be able to meet its liveness assumptions no matter how the other modules behave. For example, physical realizability, assume-guarantee reasoning about live trace inclusion, and controller synthesis for live trace inclusion all depend on the receptiveness condition. In a real-time world, liveness is automatically present in the form of diverging time. The receptiveness condition, then, translates to the requirement that a module must be able to let time diverge no matter how the environment behaves. We study the receptiveness condition for real-time systems by extending the model of Reactive Modules to timed and hybrid modules. We define the receptiveness of such a module as the existence of a winning strategy in a game of the module against its environment. By solving the game on region graphs, we present an (optimal) EXPTIME algorithm for checking the receptiveness of propositional timed modules. By giving a fixpoint characterization of the game, we present a symbolic procedure for checking the receptiveness of linear hybrid modules. Finally, we present an assume-guarantee principle for reasoning about timed and hybrid modules, and a method for synthesizing receptive controllers of timed and hybrid modules.